|  |  |
| --- | --- |
| **Une image contenant texte, clipart  Description générée automatiquement** | **Authentification des utilisateurs** |

# Présentation

L’authentification HTTP de base est relativement simple à mettre en pratique pour qui connait un peu le serveur web Apache2.

Il existe plusieurs façons de générer des fenêtres avec formulaire d'authentification mais il existe également une solution simple soit htpasswd.   
htpasswd est un utilitaire dont la fonction est de stocker les mots de passe sous une forme chiffrée à utiliser par Apache dans les services d'authentification. Son utilisation permet d’accéder à certains répertoires ou fichiers uniquement à des utilisateurs identifiés.

# Authentification de base et Authentification Digest

Il existe de nombreux **schémas d'authentification disponibles**.

Ce tableau illustre la compatibilité des différents schémas d'authentification sur différents systèmes d'exploitation. Notez que si on utilise **HTTPS**, on suppose que le **certificat SSL est valide** (non auto-signé).

|  |  |  |
| --- | --- | --- |
|  | HTTP | HTTPS |
| Authentification de base | MacOS et Linux | MacOS, Linux et Windows |
| Authentification Digest | MacOS, Linux et Windows | MacOS, Linux et Windows |

**Authentification Digest**

Dans un premier temps il faut **générer le fichier** (appelé **users.password**) qui **emmagasine les mots de passe pour les utilisateurs**.   
Dans l'authentification Digest, il existe le **champ domaine qui agit comme un espace de noms pour les utilisateurs**.

Pour cet exemple, le premier utilisateur s'appellera tux.

Pour générer le fichier digest, il faut **installer les dépendances pour Apache** …  
**>>** sudo apt install apache2-utils

# Création des utilisateurs

On doit, dans un premier temps, créer un fichier d'utilisateurs.   
Ce fichier doit se trouver dans un répertoire lisible par nginx (par exemple le répertoire dans lequel se trouve le site – le répertoire qui est servi par nginx) …  
**>>** sudo **htpasswd -c <Nom de la base de données> <Nom de l’utilisateur>  
>>** sudo htpasswd -c /var/www/html/users.password tux

L'utilitaire va demander le mot de passe pour l’utilisateur.

**Remarques** …  
Si jamais on souhaite ajouter un utilisateur, on utilisera la même commande sans l'option -c.   
Pour en supprimer un ce sera l'option -D qui sera à utiliser.  
Pour afficher la liste des utilisateurs créés …  
**>>** sudo cat /var/www/html/users.password

## Configuration de nginx

La prochaine étape est la modification du fichier de configuration du site par défaut présent dans le répertoire etc/nginx/sites-available.

Au niveau de la configuration du répertoire que l’on souhaite protéger, on ajoute les lignes auth\_basic "Restricted"; et auth\_basic\_user\_file /var/www/html/.htpasswd; …  
**>>** sudo nano /etc/nginx/sites-available/default  
**server {  
 listen 80 default\_server;  
 listen [::]:80 default\_server;  
  
 root /var/www/html;  
  
 # Add index.php to the list if you are using PHP  
 index index.html index.htm index.nginx-debian.html;**

**server\_name \_;  
  
[...]  
  
 location / {  
 auth\_basic "Restricted";  
 auth\_basic\_user\_file /var/www/html/users.password;   
 try\_files $uri $uri/ =404; }  
 [...]  
}**

Après avoir effectué une modification de configuration sur un serveur nginx, on peut lancer la commande suivante afin de s’assurer qu’il n’y a pas d’erreur …  
**>>** sudo nginx -t

Il ne reste qu’à recharger nginx …  
**>>** sudo systemctl restart nginx.service

Une image contenant texte

Description générée automatiquement

# Authentification de base et restriction d'accès par adresse IP

L'authentification de base HTTP peut être efficacement combinée avec une restriction d'accès par adresse IP.

On peut mettre en œuvre au moins deux scénarios …

* un utilisateur doit être à la fois authentifié et avoir une adresse IP valide ;
* un utilisateur doit être soit authentifié, soit avoir une adresse IP valide.

On peut autoriser ou refuser l'accès à partir d'adresses IP particulières avec les directives allow et deny …  
**>>** sudo nano /etc/nginx/sites-available/default  
**server {  
 listen 80 default\_server;  
 listen [::]:80 default\_server;  
  
 root /var/www/html;  
  
 # Add index.php to the list if you are using PHP  
 index index.html index.htm index.nginx-debian.html;**

**server\_name \_;  
  
[...]  
  
 location / {  
 auth\_basic "Restricted";  
 auth\_basic\_user\_file /var/www/html/.htpasswd;   
 satisfy all;   
  
 deny 192.168.1.2;  
 allow 192.168.1.0/24;  
 allow 127.0.0.1;  
 deny all;  
 try\_files $uri $uri/ =404; }  
 [...]  
}**

L'accès sera accordé uniquement pour le réseau 192.168.1.1/24 à l'exclusion de l'adresse 192.168.1.2.

**Remarque** …  
Les directives allow et deny seront appliquées dans l'ordre dans lequel elles sont définies.

On peut donc combine la restriction par authentification IP et HTTP avec la directive satisfy.   
Si on définit la directive sur all, l'accès est accordé si un client remplit les deux conditions.   
Si on définit la directive sur any, l'accès est accordé si un client satisfait au moins une condition.